Tutorial Discipulu's

Release 20.1.0

Jul 30, 2020

Contents

1 Módulo Main **3**
- 1.1 Neste tutorial encontrá: . . . 3
- 1.2 Classes neste módulo: . . . 3
- 1.3 Changelog . . . 3

2 Módulo Bottle_app: Gerenciador HTTP **5**

3 Módulo Delta **7**
- 3.1 Changelog . . . 7

4 Módulo Vitollino **9**
- 4.1 Gerador de labirintos e jogos tipo *'novel'*. . . . 9

5 Documentação Adicional **17**
- 5.1 **Relatable Topics**! . . . 17
- 5.2 **Comandos para SHELL Linux**! . . . 20

Python Module Index **23**

Index **25**

Author Emanuelle Simas <ellesimas@gmail.com>

Git Hub ellesimas.

CHAPTER 1

Módulo Main

New in version 20.1.0: Versão inicial.

Note: Este documento faz parte do método de Ensino da programação e desenvolvimento orientando pelo Doutor e Professor Carlo Emmanoel Tolla de Oliveira. Git Hub: CarloTolla.

1.1 Neste tutorial encontrá:

1. Uso Enconding utf 8
2. Comandos básicos no terminal linux
3. Comando de articulação pythonanywhere e github
4. Processo de criação de uma WebPage inicial
5. Instalação e uso do sphinx
6. Implementação da documentação no read the docs
7. Importação de itens externos

1.2 Classes neste módulo:

:py:class: 'Main' Exemplo de classe qualquer

1.3 Changelog

New in version 20.1.0: Documentação do tutorial

New in version 20.1.1: Chama game criado externamente

class main.**Main**(*versao='20.1.0'*)

Classe exemplo

Parameters

- **self** – Chama ele mesmo.
- **versao** – Retorna a versão colocada.

pega_versao()

Returns A versao do sistema

CHAPTER 2

Módulo Bottle_app: Gerenciador HTTP

- **As rotas que direcionam para outras abas da web,** através do gerenciador de de http.
- Framework Bottle utilizado. Documentação em <https://wiki.python.org/moin/Routing>

New in version 20.1.0.

bottle_app.**py_rota**(*filename*)
Roteia

bottle_app.**rota_credits**()
Roteia o caminho para retornar a versão do sistema. O módulo chamado é o main com sua classe Main importado linhas acima.

bottle_app.**rota_css_doc**(*filename*)
Roteia o caminho /vs para a page html gerada pelo sphinx, chamando css

bottle_app.**rota_doc**(*filename*)
Roteia o caminho /vs para a page html gerada pelo sphinx. Na web <domínio.com/doc/filename/html> Atenção que o doc da chamada deve ser o root não a pasta.

bottle_app.**rota_dois**()
Exemplo da geração de uma segunda rota qualquer.

bottle_app.**rota_game**()
Serve ao index.html que serve ao brython que rodará o vitollino no delta.py

CHAPTER 3

Módulo Delta

Note: Jogo de memória é trazido do super python

Delta - Projeto sem descrição, (mude esta linha).

3.1 Changelog

New in version 20.07: Descreva o que você adicionou no código.

class delta.**start**
Gera uma classe onde há a tela de partida do jogo da memória.

solucao = None
O bind liga um evento a um método

vai()

CHAPTER 4

Módulo Vitollino

Note: Engenho de jogo so superpython Importado no módulo anterior

4.1 Gerador de labirintos e jogos tipo *'novel'*.

Gerador de labirintos e jogos tipo *'novel'*.

Vitollino em Github

class vitollino.main.**Bloco**

conta_pecas(*valor_peca*)

inicia_de_novo()

nao_monta()

vai()

class vitollino.main.**Cena**(*img=''*, *esquerda=<CenaNula>*, *direita=<CenaNula>*, *meio=<CenaNula>*, *vai=None*, *nome=''*, *xy=(0, 0)*, *score={}*, ***kwargs*)

Use para construir uma cena.

```
from _spy.vitollino import Cena

cena_esq = Cena(img="esq.jpg")
cena_mei = Cena(img="mei.jpg", cena_esq)
cena_mei.vai()
```

Parameters

- **img** (*str*) – URL da imagem

- **esquerda** (Cena) – Cena que está à esquerda desta
- **direita** (Cena) – Cena que está à direita desta
- **meio** (Cena) – Cena que está à frente desta
- **vai** – Função a ser chamada no lugar da self.vai nativa

bota(*nome_item*)

static c(***cenas*)

portal(*esquerda=None*, *direita=None*, *meio=None*, ***kwargs*)

static q(*n=<CenaNula>*, *l=<CenaNula>*, *s=<CenaNula>*, *o=<CenaNula>*, *nome=''*, ***kwargs*)

static s(*n=<CenaNula>*, *l=<CenaNula>*, *s=<CenaNula>*, *o=<CenaNula>*, *nome=''*, ***kwargs*)

sai(*saida*)

score(***kwargs*)

tira(*item*)

vai(*ev=<NoEvent>*)

vai_direita(*_=0*)

vai_esquerda(*_=0*)

vai_meio(*_=0*)

class vitollino.main.**Codigo**(*codigo=''*, *topo=''*, *cena=Inventario*, *img=''*, *vai=None*, *style={}*)

Um objeto de interação que é representado por uma trecho de código em uma cena.

exemplo = Codigo(codigo="from anna import main", topo="Importando um módulo", vai=testa_codigo, style=dict(left=350, top=550, width=60))

Parameters

- **codigo** – O código de programa
- **vai** – função executada quando se clica no objeto
- **style** – dicionário com dimensões do objeto {"left": ..., "top": ..., width: ..., height: ...}
- **topo** – Texto que aparece no topo do bloco
- **cena** – cena onde o objeto vai ser colocado

class vitollino.main.**Cursor**(*alvo*, *cena=<BrythonMock name='mock.__getitem__()' id='139858295081056'>*)

class vitollino.main.**Dragger**(*dragger*)

ACTION = ''

POINTER = ''

drag_start(*ev*)

highten(*dy*)

mouse_down(*ev*)

static mouse_over(*ev*)

mouse_up(_)

no_mouse_move(*ev*)

pre_mouse_down(*ev*)

pre_mouse_move(*ev*)

static pre_mouse_over(*ev*)

pre_mouse_up(_)

widen(*dx*)

class vitollino.main.**Droppable**(*droppable*, *dropper_name=''*, *action=None*, *cursor=None*)

drag_over(*ev*)

drop(*ev*)

class vitollino.main.**Dropper**(*dropper*)

drag_start(*ev*)

mouse_over(*ev*)

class vitollino.main.**Elemento**(*img=''*, *vai=None*, *style={}*, *tit=''*, *alt=''*, *x=0*, *y=0*, *w=100*, *h=100*, *o=1*, *texto=''*, *cena=Inventario*, *score={}*, *drag=False*, *drop={}*, *tipo='100% 100%'*, ***kwargs*)

Um objeto de interação que é representado por uma imagem em uma cena.

papel = Elemento(img="papel.png", tit="caderno de notas", vai=pega_papel, style=dict(left=350, top=550, width=60))

Parameters

- **img** – URL de uma imagem
- **vai** – função executada quando se clica no objeto
- **style** – dicionário com dimensões do objeto {"left": ..., "top": ..., width: ..., height: ...}
- **tit** – Texto que aparece quando se passa o mouse sobre o objeto
- **alt** – Texto para leitores de tela
- **x** – Posição x, na horizontal a partir da esquerda do elemento na cena
- **y** – Posição y, na vertical a partir do topo do elemento na cena
- **w** – Largura em pixels do elemento na cena
- **h** – Altura em pixels do elemento na cena
- **texto** – Se fornecido, este texto vai aparecer quando se clica no elemento
- **cena** – cena alternativa onde o objeto vai ser colocado
- **score** – determina o score para este elemento
- **drag** – Se o valor for True, o o elmento será arrastável, default Faslse
- **drop** – recebe um dicionário {"a": ator} onde ator é uma função def ator(ev, nome} chamada quando "a" é arrastado cá

- **kwargs** – lista de parametros nome=URL que geram elementos com este nome e a dada imagem

do_drag(*drag=True*)

do_drop(*drop=''*)

do_score(*tit*)

drag

drag_over(*ev*)

drag_set

drag_start(*ev*)

drop(*ev*)

drop_set

foi()

img_drag_start(*ev*)

img_prevent(*ev*)

mouse_over(*ev*)

o

style

style_set

tit

x

y

class vitollino.main.**Elemento_**(*img=''*, *vai=None*, *style={}*, *tit=''*, *alt=''*, *cena=Inventario*, *score={}*, *drag=False*, *drop=''*, ***kwargs*)

Um objeto de interação que é representado por uma imagem em uma cena.

papel = Elemento(img="papel.png", tit="caderno de notas", vai=pega_papel, style=dict(left=350, top=550, width=60))

Parameters

- **img** – URL de uma imagem
- **vai** – função executada quando se clica no objeto
- **style** – dicionário com dimensões do objeto {"left": ..., "top": ..., width: ..., height: ...}
- **tit** – Texto que aparece quando se passa o mouse sobre o objeto
- **alt** – Texto para leitores de tela
- **cena** – cena alternativa onde o objeto vai ser colocado
- **score** – determina o score para este elemento
- **kwargs** – lista de parametros nome=URL que geram elementos com este nome e a dada imagem

classmethod c(***kwargs*)

entra(*cena*, *style={}*)

limbo = <BrythonMock name='mock.DIV()' id='139858295149512'>

score(***kwargs*)

class vitollino.main.**Folha**(*texto*, *ht_ml*, *tela*, *left*)

drag_start(*ev*)

mouse_over(*ev*)

class vitollino.main.**Inventario**(*tela=<BrythonMock name='mock.__getitem__()' id='139858295081056'>*)

Os objetos que estão de posse do jogador.

Parameters **tela** – Div do HTML onde o inventário será anexado

GID = 'de49ed95'

bota(*nome_item*, *item=''*, *drag=False*, *acao=None*)

Os objetos que estão de posse do jogador.

```
>>> inv.bota("uma_coisa")
>>> "uma_coisa" in inv.inventario
True
```

Parameters

- **nome_item** – uma string com o nome do item, ele será criado e colocado no inventário
- **item** – URL da imagem do item nomeado por nome_item
- **drag** – Se for True o objeto será arrastável
- **acao** – ação associada com o item nomeado quando ele é clicado

desmonta(*_=0*)

inicia()

monta(*_=0*)

mostra(*_=0*)

score(*casa*, *carta*, *move*, *ponto*, *valor*, *_level=1*)

static send(*operation*, *data*, *action=<function Inventario.<lambda>>*, *method='POST'*)

tira(*nome_item*)

class vitollino.main.**Jogo**

algo

Acessa a classe Elemento

cena

Acessa a classe Cena

nota

Acessa a classe Texto

quarto
Acessa a classe Sala

sala
Acessa a classe Salao

class vitollino.main.**Labirinto**(*c=<CenaNula>*, *n=<CenaNula>*, *l=<CenaNula>*, *s=<CenaNula>*, *o=<CenaNula>*)

lb()

static m(*cenas*)

static n(*cenas*)

class vitollino.main.**Musica**(*sound*, *loop=True*, *autoplay=True*, *sound_type='audio/mpeg'*)

class vitollino.main.**NoEv**
Representa um evento vazio.

```
>>> print(ev.x, ev.y)
-100 -100
```

class vitollino.main.**Point**(*x*, *y*)

px()

class vitollino.main.**Popup**(*cena*, *tit=''*, *txt=''*, *vai=None*, ***kwargs*)

POP = <Popup>

static d(*cena*, *tit=''*, *txt=''*)

static inicia()

vai()

class vitollino.main.**Portal**(*cena=None*, *debug_=False*, ***kwargs*)

L = {'cursor': 'e-resize', 'left': '90%', 'margin': '0%', 'min-height': '60%', 'po

N = {'cursor': 'n-resize', 'left': '20%', 'margin': '0%', 'min-height': '20%', 'po

O = {'cursor': 'w-resize', 'left': 0, 'margin': '0%', 'min-height': '60%', 'positi

PORTAIS = {'L': {'cursor': 'e-resize', 'left': '90%', 'margin': '0%', 'min-height':

S = {'bottom': 0, 'cursor': 's-resize', 'left': '20%', 'margin': '0%', 'min-height

Z = {'cursor': 'zoom-in', 'margin': '0%', 'min-height': '10%', 'position': 'absolu

p(***kwargs*)

vai(**_*)

class vitollino.main.**Sala**(*n=<CenaNula>*, *l=<CenaNula>*, *s=<CenaNula>*, *o=<CenaNula>*, *nome=''*, ***kwargs*)

static c(***cenas*)

leste

norte

oeste

p()

sul

class vitollino.main.**SalaCenaNula**
Define uma Sala ou uma Cena vazia.

```
>>> cena = Cena(SalaCenaNula())  # A próxima cena
>>> uma_cena = Cena(SalaCenaNula(), cena)  # Cena nula à esquerda, proxima no meio
>>> uma_cena.vai_esquerda()  # tenta navegar para a cena à esquerda
>>> # não vai, pois a cena é nula e não deixa que se navegue para ela
>>> print(INVENTARIO.cena == cena)
True
```

Deve ser usado quando um parâmetro requer uma cena mas não deve ter uma cena válida ali.

class vitollino.main.**Salao**(*n=<CenaNula>*, *l=<CenaNula>*, *s=<CenaNula>*, *o=<CenaNula>*, *nome=''*, ***kwargs*)

static c(***cenas*)

p()

class vitollino.main.**Suporte**(*bloco*, *ht_ml*, *tela*, *left*, *certa*)

drag_over(*ev*)

drop(*ev*)

class vitollino.main.**Texto**(*cena=<CenaNula>*, *tit=''*, *txt=''*, *foi=None*, ***kwargs*)

esconde(*ev=<NoEvent>*)

foi

mostra(*tit=''*, *txt=''*, *act=None*, ***kwargs*)

static texto(*tit=''*, *txt=''*, ***kwars*)

vai(*ev=<NoEvent>*)

vitollino.main.**main**()

vitollino.main.**parametrized**(*dec*)

vitollino.main.**singleton**(*cls_to_decorate*)
Decora um classe para ser um singleton e retornar sempre a mesma instância.

```
>>> @singleton
... class Mono:
...     def __init__(self):
...             self.x = 0
...
>>> Mono().x, Mono().x = 1, 2
>>> print(Mono().x == Mono().x, Mono().x)
True 2
```

Parameters **cls_to_decorate** – A classe para ser definida como singleton

Returns O decorador de singleton

vitollino.main.**wraps_class_to_mimic_wrapped**(*original_cls*)

Empacota uma classe decoradora para que apareça corretamente nos documentos.

```
>>> @wraps_class_to_mimic_wrapped
... class Exemplo:
...     ...
...
>>> print(Exemplo.__doc__)
Atualiza wrapper_cls para se assemelhar à classe original_cls.
```

Parameters **original_cls** – A Classe a ser empacotada

Returns O empacotador da classe

CHAPTER 5

Documentação Adicional

Note: Os documentos abaixo não relatam itens que alteram os módulos supracitados.

5.1 Relatable Topics!

..version::20.1.0

Note: Este é um pequeno documento em fase de construção que buscar ser um log para processos importantes que ocorreram durante a construção deste documento.

5.1.1 Comandos básicos no terminal linux

- **cat {opção}** |**arquivo**| {>}criar, [cat _.extens]visualizar,[CTRL+D]sair_editor.
- **cd** Alterna para diretório. {cd -]dir ant., [cd ca/mi/nho] último dir.
- :cp ../caminho/vigente . : Copia arquivo para a pasta/diretório vigente
- **-f** Força ação do console.
- **ls** Mostra todos os arquivos visíveis e diretórios internos ao vigente.
- **ls -a** Mostra, além dos arquivos visíveis, os ocultos.
- **mkdir** Cria diretório.
- **pip install <programa>** Instala o recurso.
- **python** |**arquivo**| Roda o script no console.
- **rm** |**arquivo**| Deleta arquivo.

- **rmdir |diretório|** Deleta diretório vazio.
- **-r** Ação recursiva. Ação de árvore toda
- **sphinx-quickstart** Gera arquivos pertinentes à ação python.
- **touch** Cria novo arquivo. Necessita extensão.
- **vi |arquivo|** Mostra o que gerará a documentação. Para sair :wq:.
- **mv <caminho/raiz> <des/ti/no>** move arquivo para outro diretório

5.1.2 Comando de articulação pythonanywhere e github

- **git clone "url projeto"** clona projeto do github.
- **git status** informa situação do repositório vigente (se existente).
- **git add <file>** Adiciona o documento no mester.
- **git commit -am 'mensagem'** Atualiza as informações no projeto github.
- **git push** Envia conteúdo da IDE para o gestor.
- **git pull** Upa conteúdo do gestor para a IDE.

5.1.3 Erros

- **Non-ASCII 'Ã'** Parte do código em pt não reconhecido. #-- *coding: utf-8* --.
- **No module named 'bottle'** O readthedocs não encontrou o requerimento bottle. Crie um através de um .txt!.
- ***** No rule to make target 'html'. Stop.** Dirija-se ao diretório que apresente o 'makefile'.
- **sphinx.errors.SphinxError** master file /home/../contents.rst not found: master_doc = 'index' no conf.py.
- :brython.js:5218 Uncaught vitollino: Ao rodar o web page o vitollino não foi encontrado por ausênci do _init_. Gere um _init_ no src do seu repositório.
- **uncaught delta error 5218** Com certeza é erro de sintaxe no módulo que está tentando importar! vá consertar!
- **WARNING** autodoc: failed to import module 'delta'; the following exception was raised: Este erro corrobora com a impossibilidade do sphinx importa o que o módulo importa
- **No module named 'browser'** Ele procurou o módulo requerido e não o encontrou. Uma solução é gerar um pythom mínimo dentre a sua documentação. Importe, copie ou mova o mockbrythom para o seu diretório principal
- : WARNING: Unknown directive type "doctest": Doctest é uma extensão do sphinx requerida em uma das diretivas do vitollino. Logo, insira-o no conf.py

5.1.4 Importação de diretórios internos

1. Arquivos importados para o diretório externo ao repositório vigente não aparece na documentação.
2. Faz-se necessário IMPORTAR as dependências!

5.1.5 Instalação E Uso Do Sphinx

- **Sphinx** É o gerenciador de documentação python. Ele media a transferência para o Read the docs.
- **sphinx-quickstart** É encontrado na documentação sphinx como o disparador do processo de documentação.
- **make html** The HTML pages are in build/html. USAR SEMPRE QUE ADICIONAR ALGO QUE TENHA DE SER LIDO EM HTML.
- **Diretivas**
- **toctree** Interno ao rst.py. O toctree é uma diretiva raiz da árvore de índice. É uma maneira de conectar vários arquivos numa única hierarquia.
- **automodule**
- **Extensions**

5.1.6 Operações Python

- **Format()**
- **mock** Unittest.mocké uma biblioteca para teste em Python. Ele permite que você substitua partes do seu sistema em teste por objetos simulados e faça afirmações sobre como elas foram usadas.
- **pylint** O Pylint é uma ferramenta de análise de código estática do Python que procura erros de programação, ajuda a impor um padrão de codificação, fareja cheiros de código e oferece sugestões simples de refatoração.

5.1.7 Organização Dos Diretórios

- **dev** Diretório principal de desenvolvimento;
- **Source |src|** Diretório para acomodação de arquivos .py;
- **docs** Diretório para acomodação dos arquivos de documentação;
- **docs/source/conf.py** É onde estarão as principais diretrizes de organização do documento;
- **docs/source/index.rst** Contém a raiz da árvore de índice ou toctree;

5.1.8 Organização Dos Arquivos

- :index:Geralmente é uma página principal
- **html** Serve as informações antes compostas de python
- **index.html**

5.1.9 WEB

- **html| mimetype|** O MIME type é o mecanismo para dizer ao cliente a variedade de documentos transmitidos. Sintaxe - Tipo/subtipo

5.1.10 HTML

- **index.html** Documento principal que servirá os documentos importados. Necessário fazer para cada nova sala?
- **head** Instruções preliminares para antes de implementar o html
- **body** É o corpo de funcionamento do html a ser rodado.
- **<script type="text/python">** Diz que a partir desta linha o que está escrito é python

5.1.11 CSS

- **viewport** Auxilia css a redimensionar a tela quando janela for maximizada ou minimizada
  - **content="width=device-width, initial-scale=1"** Repassa para o ccs a largura do dispositivo para poder ajustar o Layout

5.1.12 BRYTHON

- **Brython** É o interpretador python escrito em javascript. É um compilador que permite rodar o python dentro do browser.
- **cdn** É um endereço que permite pegar um objeto do servidor mais próximo à minha localização.
- <body onload="brython()">: Carregador visual

5.1.13 VITOLLINO

- ** ** O vitollino procura no corpo de html cujo id seja pydiv, colocando todos os componentes do vitollino.
- **div** É um container do html. Uma caixinha.

5.2 Comandos para SHELL Linux!

5.2.1 Introdução ao Shell

- **~___/__/__$** Intrui sobre onde você está no bash.
- **pwd** Indica em qual repositório o usuário está.
- **hostname** apresenta o nome do host local.
- **Shel** Interpreta os comando do terminal.
- **man <comando>** é o manual de comandos. 'q' exita.

5.2.2 Mudança de Diretório

1. **cd /nome_do_diretório** direciona para o diretório requerido;
2. **cd ~** volta para o diretório raiz;
3. **cd -** volta para o diretório inicial, home;

5.2.3 Mostrador

1. **ls** Diz os arquivos dentro do diretório;
2. **ls -a** Mostra os ocultos;
3. **ls <nome do diretório>** Mostra os arquivos da pasta pedida;
4. **ls -F** Os diretórios são apresentados como 'dir/';
5. **ls -F /etc** Informa os diretórios do destino 'int_etc/';
6. **ls -l <caminho>** Mostra em lista longa o Tipo de arquivo e adicionais;
7. **file <arquivo>** Diz o tipo de arquivo text, executável, etc.;

- **-** Arquivo comum;
- **d** Diretório;
- **b** Arquivo de bloco. Referente a gerenciadores de bloco;

1. **cat <nome_do_arquivo>** Mostra o conteúdo do arquivo;
2. **cat > <nome_do_arquivo>** Permite a escrita no editor do SHELL.CTRL+C exita;

5.2.4 Criação e destruição

1. **mkdir <nome_desejado_>** Criação de diretório;
2. **rmdir <nome_diretorio>** Apaga diretórios VAZIOS;
3. **rm -r <nome_diretorios>** Apaga diretorios com elementos em recursão;
4. **rm -fr <nome_diretorio>** Força o apagamento do diretório;
5. **rm -i <nome_do_arquivo.ext>** Apaga arquivo solcitando confirmação;
6. **echo "Texto 1" > <nome_e_extensão_a_ser_criado>** criar um arquivo com a extensão e conteúdo colocados;
7. **echo "Texto 2" >> <nome_e_extensão_a_ser_criado>** Adiciona o novo conteúdo abaixo do conteúdo anterior;
8. **cp <nome_arquivo.ext>** Cria uma cópia do arquivo e diretórios ao qual é gerado no mesmo lugar da origem;
9. **cd -avr <nome_do_arquivo_de_transferencia>**

- Copia mantendo os atrivbutos <-a>, permite visualização dos processos;
- <-v> e pratica em recursão;
- <r> no arquivo de destino;

1. **cat > <nome_do_arquivo>** Permite criação e escrita no editor do SHELL. CTRL+C exita;
2. **touch <caminho/nome_do_arquivo.txt>** Cria arquivo;
3. **touch <caminh/nome_do_arquivo{1..2}.txt>** Cria duas versões do arquivo;

5.2.5 Alterar

1. **mv <nome_atual.ext> <nome_requerido.ext>** Altera nome do arquivo;
2. **cat > <nome_do_arquivo>** Permite a escrita no editor do SHELL. para sair, CTRL+C;

5.2.6 Movientação

1. **mv <arquivo.ext> <caminho/de/destino>** move o arquivo para o caminho especificado;
2. **mv Caminho/requerido/ <nome_original> <nome_requerido>** Altera a localização do arquivo e muda nome;

5.2.7 Especificações

1. **inicio_requerido + *** "*" é completado por qualquer informação que COMECE por nome_requerido;

Python Module Index

b

bottle_app (*Web*), 5

c

Comandos_Linux (*Web*), 20

d

delta (*Web*), 7

i

Intern_Description (*Web*), 17

m

main (*Web*), 3

v

Vitollino (*Web*), 9
vitollino.main (*Web*), 9

Index

A

ACTION (*vitollino.main.Dragger attribute*), 10
algo (*vitollino.main.Jogo attribute*), 13

B

Bloco (*class in vitollino.main*), 9
bota() (*vitollino.main.Cena method*), 10
bota() (*vitollino.main.Inventario method*), 13
bottle_app (*module*), 5

C

c() (*vitollino.main.Cena static method*), 10
c() (*vitollino.main.Elemento_ class method*), 12
c() (*vitollino.main.Sala static method*), 14
c() (*vitollino.main.Salao static method*), 15
Cena (*class in vitollino.main*), 9
cena (*vitollino.main.Jogo attribute*), 13
Codigo (*class in vitollino.main*), 10
Comandos_Linux (*module*), 20
conta_pecas() (*vitollino.main.Bloco method*), 9
Cursor (*class in vitollino.main*), 10

D

d() (*vitollino.main.Popup static method*), 14
delta (*module*), 7
desmonta() (*vitollino.main.Inventario method*), 13
do_drag() (*vitollino.main.Elemento method*), 12
do_drop() (*vitollino.main.Elemento method*), 12
do_score() (*vitollino.main.Elemento method*), 12
drag (*vitollino.main.Elemento attribute*), 12
drag_over() (*vitollino.main.Droppable method*), 11
drag_over() (*vitollino.main.Elemento method*), 12
drag_over() (*vitollino.main.Suporte method*), 15
drag_set (*vitollino.main.Elemento attribute*), 12
drag_start() (*vitollino.main.Dragger method*), 10
drag_start() (*vitollino.main.Dropper method*), 11
drag_start() (*vitollino.main.Elemento method*), 12
drag_start() (*vitollino.main.Folha method*), 13
Dragger (*class in vitollino.main*), 10
drop() (*vitollino.main.Droppable method*), 11
drop() (*vitollino.main.Elemento method*), 12
drop() (*vitollino.main.Suporte method*), 15
drop_set (*vitollino.main.Elemento attribute*), 12
Droppable (*class in vitollino.main*), 11
Dropper (*class in vitollino.main*), 11

E

Elemento (*class in vitollino.main*), 11
Elemento_ (*class in vitollino.main*), 12
entra() (*vitollino.main.Elemento_ method*), 13
esconde() (*vitollino.main.Texto method*), 15

F

foi (*vitollino.main.Texto attribute*), 15
foi() (*vitollino.main.Elemento method*), 12
Folha (*class in vitollino.main*), 13

G

GID (*vitollino.main.Inventario attribute*), 13

H

highten() (*vitollino.main.Dragger method*), 10

I

img_drag_start() (*vitollino.main.Elemento method*), 12
img_prevent() (*vitollino.main.Elemento method*), 12
inicia() (*vitollino.main.Inventario method*), 13
inicia() (*vitollino.main.Popup static method*), 14
inicia_de_novo() (*vitollino.main.Bloco method*), 9
Intern_Description (*module*), 17
Inventario (*class in vitollino.main*), 13

J

Jogo (*class in vitollino.main*), 13

L

L (*vitollino.main.Portal attribute*), 14

Labirinto (*class in vitollino.main*), 14
lb() (*vitollino.main.Labirinto method*), 14
leste (*vitollino.main.Sala attribute*), 14
limbo (*vitollino.main.Elemento_ attribute*), 13

M

m() (*vitollino.main.Labirinto static method*), 14
Main (*class in main*), 4
main (*module*), 3
main() (*in module vitollino.main*), 15
monta() (*vitollino.main.Inventario method*), 13
mostra() (*vitollino.main.Inventario method*), 13
mostra() (*vitollino.main.Texto method*), 15
mouse_down() (*vitollino.main.Dragger method*), 10
mouse_over() (*vitollino.main.Dragger static method*), 10
mouse_over() (*vitollino.main.Dropper method*), 11
mouse_over() (*vitollino.main.Elemento method*), 12
mouse_over() (*vitollino.main.Folha method*), 13
mouse_up() (*vitollino.main.Dragger method*), 10
Musica (*class in vitollino.main*), 14

N

N (*vitollino.main.Portal attribute*), 14
n() (*vitollino.main.Labirinto static method*), 14
nao_monta() (*vitollino.main.Bloco method*), 9
no_mouse_move() (*vitollino.main.Dragger method*), 11
NoEv (*class in vitollino.main*), 14
norte (*vitollino.main.Sala attribute*), 14
nota (*vitollino.main.Jogo attribute*), 13

O

o (*vitollino.main.Elemento attribute*), 12
O (*vitollino.main.Portal attribute*), 14
oeste (*vitollino.main.Sala attribute*), 14

P

p() (*vitollino.main.Portal method*), 14
p() (*vitollino.main.Sala method*), 15
p() (*vitollino.main.Salao method*), 15
parametrized() (*in module vitollino.main*), 15
pega_versao() (*main.Main method*), 4
Point (*class in vitollino.main*), 14
POINTER (*vitollino.main.Dragger attribute*), 10
POP (*vitollino.main.Popup attribute*), 14
Popup (*class in vitollino.main*), 14
PORTAIS (*vitollino.main.Portal attribute*), 14
Portal (*class in vitollino.main*), 14
portal() (*vitollino.main.Cena method*), 10
pre_mouse_down() (*vitollino.main.Dragger method*), 11
pre_mouse_move() (*vitollino.main.Dragger method*), 11
pre_mouse_over() (*vitollino.main.Dragger static method*), 11
pre_mouse_up() (*vitollino.main.Dragger method*), 11
px() (*vitollino.main.Point method*), 14
py_rota() (*in module bottle_app*), 5

Q

q() (*vitollino.main.Cena static method*), 10
quarto (*vitollino.main.Jogo attribute*), 13

R

rota_credits() (*in module bottle_app*), 5
rota_css_doc() (*in module bottle_app*), 5
rota_doc() (*in module bottle_app*), 5
rota_dois() (*in module bottle_app*), 5
rota_game() (*in module bottle_app*), 5

S

S (*vitollino.main.Portal attribute*), 14
s() (*vitollino.main.Cena static method*), 10
sai() (*vitollino.main.Cena method*), 10
Sala (*class in vitollino.main*), 14
sala (*vitollino.main.Jogo attribute*), 14
SalaCenaNula (*class in vitollino.main*), 15
Salao (*class in vitollino.main*), 15
score() (*vitollino.main.Cena method*), 10
score() (*vitollino.main.Elemento_ method*), 13
score() (*vitollino.main.Inventario method*), 13
send() (*vitollino.main.Inventario static method*), 13
singleton() (*in module vitollino.main*), 15
solucao (*delta.start attribute*), 7
start (*class in delta*), 7
style (*vitollino.main.Elemento attribute*), 12
style_set (*vitollino.main.Elemento attribute*), 12
sul (*vitollino.main.Sala attribute*), 15
Suporte (*class in vitollino.main*), 15

T

Texto (*class in vitollino.main*), 15
texto() (*vitollino.main.Texto static method*), 15
tira() (*vitollino.main.Cena method*), 10
tira() (*vitollino.main.Inventario method*), 13
tit (*vitollino.main.Elemento attribute*), 12

V

vai() (*delta.start method*), 7
vai() (*vitollino.main.Bloco method*), 9
vai() (*vitollino.main.Cena method*), 10
vai() (*vitollino.main.Popup method*), 14
vai() (*vitollino.main.Portal method*), 14
vai() (*vitollino.main.Texto method*), 15
vai_direita() (*vitollino.main.Cena method*), 10

vai_esquerda() (*vitollino.main.Cena method*), 10
vai_meio() (*vitollino.main.Cena method*), 10
Vitollino (*module*), 9
vitollino.main (*module*), 9

W

widen() (*vitollino.main.Dragger method*), 11
wraps_class_to_mimic_wrapped() (*in module vitollino.main*), 16

X

x (*vitollino.main.Elemento attribute*), 12

Y

y (*vitollino.main.Elemento attribute*), 12

Z

Z (*vitollino.main.Portal attribute*), 14